**Spark Setup using Docker:  
  
Step 1: Install Docker**

**For Windows:**

1. Download Docker Desktop from the official Docker website.
2. Follow the installation instructions.
3. After installation, open Docker Desktop and ensure it is running. Docker should be visible in the system tray.

**For Ubuntu:**

1. Install Docker by running the following commands in your terminal:

sudo apt update

sudo apt install docker.io

sudo systemctl start docker

sudo systemctl enable docker

1. Verify Docker installation:

docker --version

1. Add your user to the Docker group (optional for running Docker without sudo):

sudo usermod -aG docker $USER

Make sure Docker is installed on your system. You can follow the installation guide from Docker's official site.

**Step 2: Pull the Bitnami Spark Docker Image**

Pull the Bitnami Spark Docker image to your local machine:

docker pull bitnami/spark

**Step 3: Create Your PySpark Code**

Create your PySpark script (e.g., app.py). You can place it in a directory, say ~/spark-app.

**Step 4: Prepare the Directory for Your Code**

**For Windows:**

1. Create the directory for your Spark application:

mkdir C:\spark-app

1. Move the app.py script into this directory:

mv C:\path\to\app.py C:\spark-app\

**For Ubuntu:**

1. Create the directory for your Spark application:

mkdir ~/spark-app

1. Move the app.py script into this directory:

mv /path/to/app.py ~/spark-app/

**Step 5: Run the Docker Container**

Run the Bitnami Spark Docker container, mapping necessary ports and volumes:

docker run -it --rm -p 8888:8888 -p 4040:4040 -v C:\spark-app:/app bitnami/spark

**Step 6: Submit Your PySpark Job**

Find the container ID of the running Spark container:

docker ps

Then, execute the PySpark job inside the container:

docker exec -it <container-id> /opt/bitnami/spark/bin/spark-submit /app/app.py

**Use Case: Log File Analysis for System Health Monitoring**

**Scenario:**

You are tasked with analyzing server logs collected daily in the following format: /data/logs/YYYY-MM-DD/\*.log. Each log entry contains the following fields:

* timestamp: The date and time of the log entry.
* server\_id: The unique identifier of the server.
* log\_level: The severity of the log (e.g., INFO, WARN, ERROR).
* message: The log message content.

Your tasks are:

1. Identify the **top 3 servers** with the highest number of ERROR logs over the past week.
2. Calculate the **average number of logs generated per day** by each server over the past week.
3. Provide a **summary report** of the most common log messages for each severity level.